Rancang Bangun Aplikasi Deteksi Kesalahan Penulisan Naskah Dokumen Skripsi

Wisnu W. A. Umboh, Steven R. Sentinuwo, Alwin M. Sambul
Teknik Informatika, Universitas Sam Ratulangi Manado, Indonesia.
120216009@student.unsrat.ac.id, steven@unsrat.ac.id, asambul@unsrat.ac.id

***Abstrak* – Dokumen skripsi adalah salah satu syarat agar mahasiswa dapat menyelesaikan program studi di sebuah universitas. Namun pada kenyataanya mahasiswa masih sering melakukan kesalahan dalam penulisan naskah dokumen skripsi. Aplikasi deteksi kesalahan penulisan skripsi merupakan solusi untuk membantu mahasiswa dalam membuat skripsi dan mendeteksi kesalahan penulisan dokumen skripsi. Salah satu metode *indexing* untuk meng-indeks teks biasa, untuk mengurangi kapasitas pemakaian storage dan meningkatkan kinerja *searching* adalah *Full Text Indexing*. *Full Text Indexing* merupakan metode yang digunakan dalam mencari kesalahan dalam sebuah teks sebagai alat bantu utama dalam perancangan aplikasi ini. Pada metode *Full Text Indexing* terdapat 2 tahap yang dilakukan sebelum dilakukan pencarian kata, yaitu tahap *tokenizing* dan tahap cleansing. Aplikasi deteksi kesalahan penulisan naskah dokumen skripsi dibuat dengan fitur pengecekan kesalahan penulisan dan penyimpanan daftar pustaka dan daftar gambar. Selain itu, aplikasi ini memiliki fitur *preview* sebagai fitur untuk menampilkan hasil data yang disimpan pada bab 1 sampai 5, dan juga menampilkan daftar pustaka yang telah disimpan. Dengan dibuatnya aplikasi ini diharapkan agar aplikasi ini bisa membantu mahasiswa dalam pembuatan skripsi, terutama dalam pengecekan kesalahan penulisan skripsi.**

Kata Kunci : Aplikasi, Deteksi, Kesalahan, Penulisan, Skripsi, *Full*, *Text*, *Indexing*

I. PENDAHULUAN

Komputer berperan penting dalam mempermudah pekerjaan sehari-hari. Salah satu manfaat utama komputer adalah sebagai alat bantu untuk membuat karya tulis. Berbagai aplikasi seperti *Microsoft Word*, *Notepad*, maupun *OpenOffice Word* biasanya terdapat pada komputer untuk mempermudah pengguna. [2]

Skripsi pada mahasiswa di Universitas Sam Ratulangi pada umumnya dapat membawa manfaat yang sangat positif bagi pengembangan kurikulum maupun perencanaan roadmap penelitian skala institusi, terutama pada Program Studi Informatika Fakultas Teknik Unsrat.

Namun, kesalahan penulisan dalam dokumen skripsi masih saja dapat ditemukan. Kesalahan-kesalahan yang umumnya terjadi antara lain: penggantian satu huruf, penyisipan satu huruf, penghilangan satu huruf, maupun penukaran dua huruf berdekatan.

Dr. Tom Stafford, peneliti di bidang psikologi dan ilmu kognitif manusia yang juga menjadi pengajar di Universitas Sheffield, Inggris telah melakukan penelitian secara khusus tentang atau kekeliruan dalam sebuah tulisan bahwa kegiatan menulis memaksa seseorang untuk secara bersamaan melihat segalanya sebagai percampuran antara data pasti yang diterima sensor indera dan ekspektasi kita akan suatu hal. Sederhananya, kita sulit menyadari adanya suatu kesalahan karena pikiran kita telah terpaku pada bayangan ideal tentang apa yang kita tulis. [6]

Untuk mengatasi masalah tersebut, diperlukan suatu aplikasi yang dapat digunakan untuk mempermudah pengguna dalam memeriksa adanya kesalahan ejaan dalam laporan, sekaligus mendeteksi penempatan tanda baca yang salah pada dokumen skripsi. Melalui penelitian ini penulis merancang dan membangun sebuah aplikasi untuk deteksi kesalahan penulisan pada skripsi mahasiswa. Proses tersebut dilakukan secara semi-otomatis dengan memanfaatkan teknologi word preprocessor dan aplikasi ini akan dibangun berbasiskan web.

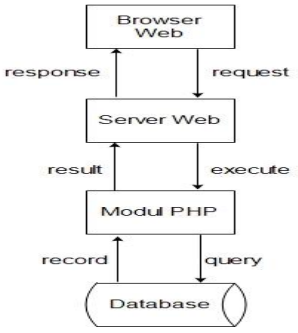
II. LANDASAN TEORI

A. Konsep Dasar Aplikasi *Web*

Menurut Fauzi Rahman (2013), aplikasi adalah sekelompok atribut yang terdiri dari beberapa *form*, yang disusun sedemikian rupa sehingga dapat mengakses data. Aplikasi merupakan program yang berisikan perintah-perintah untuk melakukan pengolahan data. [7]

Berdasarkan pengertian tersebut, dapat disimpulkan bahwa aplikasi merupakan *software* yang ditransformasikan ke komputer yang berisikan perintah-perintah yang berfungsi untuk melakukan berbagai bentuk pekerjaan atau tugas-tugas tertentu seperti penerapan, penggunaan dan penambahan data.

Web berasal dari kata Bahasa Inggris yang bila diterjemahkan dalam Bahasa Indonesia berarti "Jaring Laba-Laba". Web telah membentang ke seluruh penjuru dunia. Tidak hanya terbatas pada lembaga-lembaga penelitian yang ingin memublikasikan hasil riset, tetapi juga telah banyak digunakan oleh perusahaan bisnis yang ingin mengiklankan produk atau untuk melakukan transaksi bisnisnya. [8]



Gambar 1. Diagram Aplikasi *Web*.

B. PHP

PHP adalah kependekan dari PHP *Hypertext Preprocessor*, bahasa interpreter yang mempunyai kemiripan dengan bahasa C dan Perl yang mempunyai kesederhanaan dalam perintah, yang digunakan untuk pembuatan aplikasi *web*. PHP awalnya merupakan program CGI (*Common Gateway Interface*) yang dikhususkan untuk menerima input melalui form yang ditampilkan dalam *browser web*. [7]

C. XAMPP

Dalam buku ”Pemrograman *Web* PHP”, XAMPP (X (*Windows/Linux*) Apache MySQL PHP and Perl) adalah paket *server web* PHP dan *database* MySQL yang paling populer untuk para pengembang *web* dengan menggunakan PHP dan MySQL sebagai *database*. [5]

D. MySQL

MySQL adalah *multiuser database* yang menggunakan bahasa *Structured Query Language* (SQL). MySQL dalam operasi client server melibatkan *server daemon* MySQL disisi *server* dan berbagai macam program serta *library* yang berjalan disisi *client*. MySQL mampu menangani data yang cukup besar. [3]

E. Jenis Kesalahan Penulisan

Bahasa karya ilmiah Bahasa Indonesia yang baku menurut standar ilmiah yang mengacu pada Kamus Umum Bahasa Indonesia, Ejaan Yang Disempurnakan, Pedoman Umum Pembentukan Istilah dan suplemen-suplemen terbaru. Menulis karya ilmiah yang komunikatif sangat penting agar tujuan yang ingin disampaikan dapat dipahami dengan jelas dan jernih. [1]

F. Jenis Kesalahan Penulisan Pada *Microsoft Word*

Kimberly Martin menuliskan bahwa jenis kesalahan penulisan pada *Word* hampir sama dengan jenis kesalahan penulisan karya ilmiah. Selain Kimberly, redaktur pada website www.artikelinternet.com juga mengemukakan ada beberapa jenis kesalahan yang terdapat pada pengetikan skripsi menggunakan *Microsoft Word*. [3]

III. METODOLOGI PENELITIAN

A. Objek dan Lokasi Penelitian

Penelitian ini mengambil studi kasus yang bertempat di lingkungan kampus Universitas Sam Ratulangi, Manado. Objek yang berkaitan dalam penelitian ini mengambil skripsi mahasiswa Teknik Informatika Universitas Sam Ratulangi.

B. Metode Pengumpulan Data

Metode pengumpulan data dalam penelitian ini terbagi atas :

1. Studi Pustaka
   Pada bagian ini dilakukan kajian pustaka dengan membaca buku-buku dan hasil penelitian dari beberapa peneliti sebelumnya yang memiliki hubungan dengan penelitian ini agar mendapat landasan teori mengenai masalah yang akan diteliti.
2. Observasi
   Pada bagian ini penulis melakukan observasi pada objek penelitian dengan tujuan untuk memperoleh berbagai data konkret secara langsung di lapangan atau tempat penelitian.

C. Analisis Kebutuhan

Pada bagian ini, penulis menganalisis kebutuhan dalam merancang dan membangun aplikasi. Kebutuhan tersebut adalah berupa bahan dan alat untuk melakukan penulisan kode sumber dan pembangunan *database*. Bahan dan alat yang digunakan adalah :

1) Laptop
2) Printer
3) XAMPP
4) *Notepad++*
5) *Google Chrome*

D. Perancangan Aplikasi

Untuk aplikasi antarmuka dibangun menggunakan Notepad++, dengan bahasa pemrograman PHP. Program dirancang akan menggunakan 4 (empat) buah form, yaitu :

1) Form *Login* , *form* ini berguna untuk proses menyeleksi pengguna. *User* yang boleh masuk ke *form* berikutnya hanyalah *user* yang memiliki *username* dan *password* untuk *login* ke dalam halaman selanjutnya.
2) Form *Check*, *form* ini berfungsi untuk melakukan deteksi kesalahan penulisan pada teks yang diinput atau menyimpan teks yang diinput.
3) Form Daftar Pustaka, *form* ini akan menyimpan data daftar pustaka yang

E. Perancangan Sistem

Secara garis besar, cara kerja sistem ini dapat dilihat pada gambar 2, dimana bentuk masukan data (*input*) adalah *string* kalimat, dan keluaran (*output*) adalah GUI tampilan hasil deteksi kesalahan.

Sistem ini memiliki 4 fungsi utama, yaitu fitur untuk melakukan pengecekan kesalahan, fitur untuk menyimpan teks kalimat yang diinput, fitur untuk menyimpan daftar pustaka, dan fitur untuk menyimpan daftar gambar.

Gambar 2 menjelaskan bagaimana alur kerja dari aplikasi yang akan dirancang. Pertama, user memasukkan string kalimat. Kemudian, kalimat tersebut akan dipecah-pecah menjadi kata per kata. Kata tersebut kemudian diberi sebuah variabel (misalkan j) sebagai jumlah kata pada kalimat tersebut dan variabel n = 1 untuk mencocokan apakah variabel n sudah melewati variabel j atau belum.

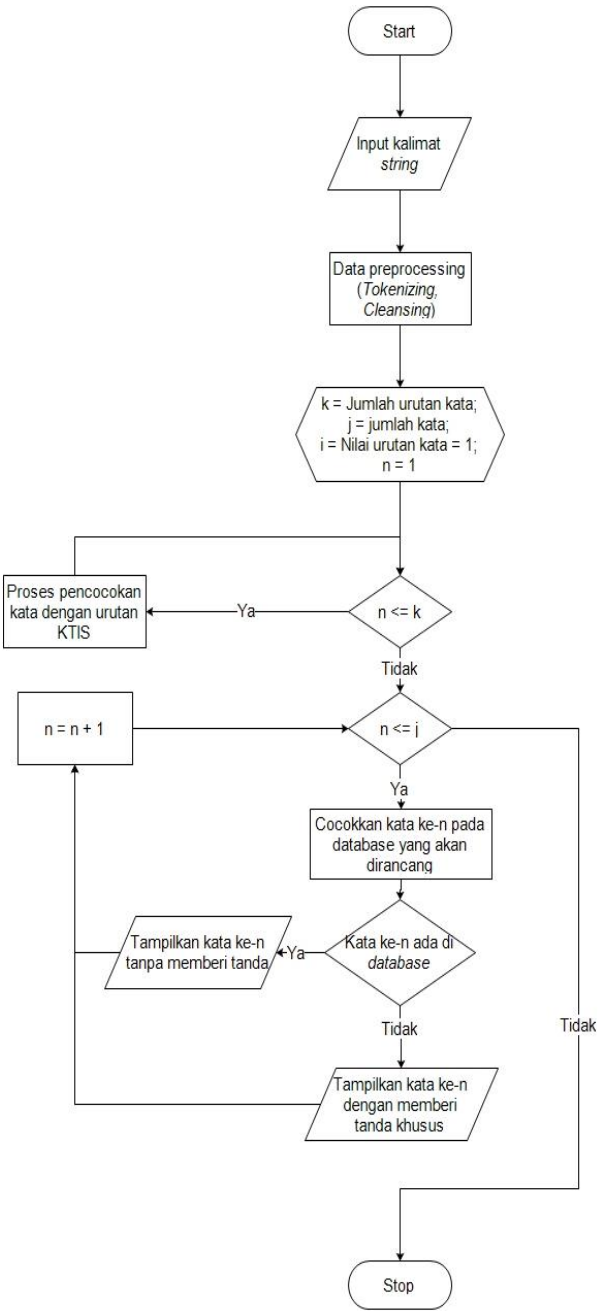
Pada bagian ini, kata-kata yang telah dipecah kemudian dicocokan dengan kata yang ada pada database. Jika kata ke-n ada di database, maka kata ke-n tersebut akan ditampilkan tanpa memberikan tanda khusus. Jika kata ke-n tidak ada di dalam database, maka kata ke-n tersebut akan diberi tanda khusus sebagai penanda bahwa kata ke-n tersebut salah.

Sesudah melakukan pencocokan kata ke-n dengan database, nilai dari variabel n kemudian ditambah 1. Jika nilai variabel n masih kurang atau sama dengan nilai variabel j, maka sistem akan kembali melakukan pencocokan kata. Jika nilai n sudah lebih dari nilai j, maka sistem akan berhenti melakukan pencocokan kata.

Sebelum melakukan pendeteksian kata dengan database, kata tersebut akan melalui proses pencocokan

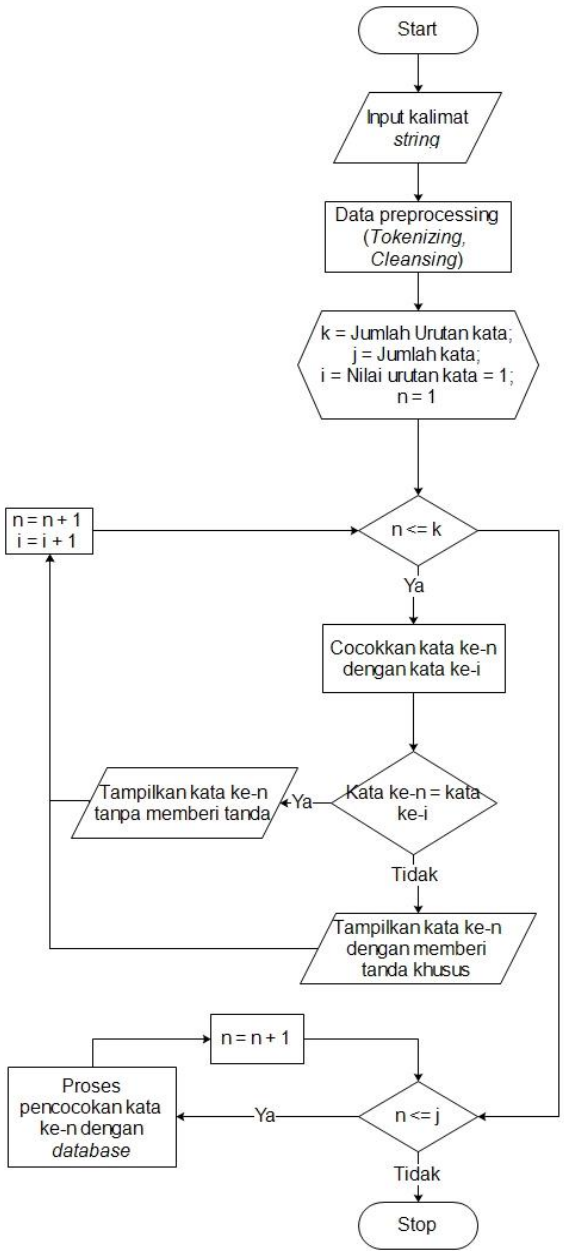
kata sesuai urutan kata pada aplikasi. Proses tersebut dapat dilihat pada gambar 3.

Alur diagram alir pada gambar 2 dan gambar 3 hampir memiliki skema yang sama. Perbedaan alur diagram alir tersebut terletak pada urutan proses deteksi kata. Pada gambar 2, menjelaskan mengenai cara kerja deteksi kata untuk dicocokkan dengan *database*, sedangkan pada gambar 3 menjelaskan mengenai cara kerja deteksi kata untuk dicocokkan dengan kata yang sudah diinisialisasi terlebih dahulu.



Gambar 2. Diagram Alir Aplikasi Pencocokkan Kata dengan *Database*.

Pada gambar 3, dilakukan pencocokan kata terlebih dahulu sesuai dengan urutan kata yang dirancang. Alur flowchart hampir sama dengan gambar 2, tetapi pada bagian penentuan variabel, ditambahkan variabel k sebagai variabel untuk penentuan jumlah urutan kata dan variabel i sebagai penentuan kata yang telah disiapkan oleh sistem sebagai pencocokan kata sesuai dengan urutan penulisan KTIS. Jika kata ke-n sama dengan kata ke-i, maka kata ke-n tersebut akan ditampilkan tanpa memberikan tanda khusus. Jika kata ke-n tidak sama dengan kata ke-i, maka kata ke-n tersebut akan diberi tanda khusus sebagai penanda bahwa kata ke-n tersebut salah. Proses tersebut diulang hingga nilai n sudah melebihi nilai k. Proses tersebut dilanjutkan dengan pendeteksian kata dengan database. Proses tersebut dapat dilihat pada gambar 2.



Gambar 3. Diagram Alir Aplikasi Pencocokkan Kata Sesuai Urutan KTIS.

F. Perancangan *Database*

Database yang digunakan dalam sistem menggunakan *database* berformat .sql, dan menggunakan PHPMyAdmin. Dibawah ini adalah penjelasan dari masing-masing tabel yang dibangun :

1) *Table Name* : logindatabase

TABEL 1
TABLE DEFINITION : LOGINDATABASE

Nama Kolom	Tipe Data	Atribut
Username	Varchar(15)	*Primary key*
Nama	Varchar(40)	
Password	Varchar(30)	

2) *Table Name* : database_konten

TABEL 2
TABLE DEFINITION : DATABASE_KONTEN

Nama Kolom	Tipe Data	Atribut
Username	Varchar(15)	*Primary key*
Bab1	Mediumtext	
Bab2	Mediumtext	
Bab3	Mediumtext	
Bab4	Mediumtext	
Bab5	Mediumtext	

3) *Table Name* : tabel_databasegambar

TABEL 3
TABLE DEFINITION : DATABASE_GAMBAR

Nama Kolom	Tipe Data	Atribut
Username	Varchar(50)	
Nomor_gambar	Varchar(10)	
Deskripsi_gambar	Varchar(100)	
Nama_gambar	Varchar(100)	
gambar	Longblob	

4) *Table Name* : database_dp

TABEL 4
TABLE DEFINITION : DATABASE_DP

Nama Kolom	Tipe Data	Atribut
Username	Varchar(25)	
jenis	Varchar(30)	
penulis	Varchar(100)	
Judul	Varchar(50)	
Judul_jurnal	Varchar(100)	
volume_jurnal	Varchar(10)	
gelar	Varchar(10)	
Link	Varchar(150)	
Tgl_publikasi	Varchar(50)	
Tgl_akses	Varchar(50)	
instansi	Varchar(100)	
penerbit	Varchar(100)	

5) *Table Name* : tb_katadasar

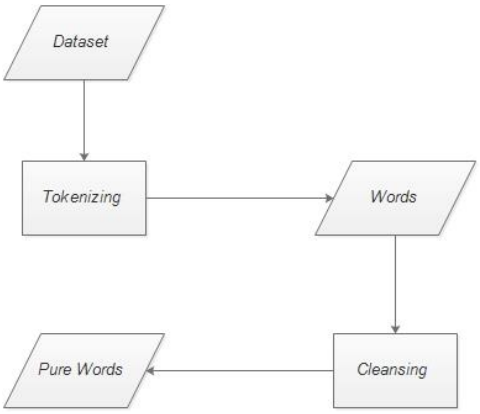
TABEL 5
TABLE DEFINITION : TB_KATADASAR

Nama Kolom	Tipe Data	Atribut
kata	Varchar(70)	*Unique key*

G. Metode Pencarian Kata

Dalam melakukan pencarian kata di dalam suatu database, digunakan metode *full text indexing*. Metode ini terdiri dari tahap *tokenizing* dan *cleansing* yaitu tahap *preprocessing* yang dimulai dari memecah kalimat menjadi kata per kata, membersihkan kata dari karakter spesial, dan mengambil setiap kata yang ada pada dataset, dalam hal ini adalah *text box* yang berisi kumpulan kalimat.

Proses pertama dari search adalah memasukkan teks kalimat pada sebuah *text box*. Kalimat yang dimasukkan pada *text box* dipecah-pecah menjadi kumpulan kata-kata, yang kemudian dijadikan sebagai *query* yang akan digunakan untuk pencarian di *database*. Jumlah *index* pada *database* yang sesuai (*total matched index*) dengan masukan pada *text box* akan digunakan untuk memunculkan hasil deteksi kesalahan. Alur proses dari metode *Full Text Indexing* ini dapat dilihat pada gambar 3.



Gambar 4. Alur *Preprocessing* Metode *Full Text Indexing*.

Gambar 4 menjelaskan cara kerja penggunaan metode *full text indexing*. Pertama, *dataset* yang dimasukkan adalah *dataset* berupa kalimat *string* panjang. *Dataset* tersebut akan melalui proses *tokenizing*, dimana kalimat akan dipecah-pecah menjadi kata per kata. Kemudian kata tersebut akan melalui proses *cleansing*, dimana kata yang mengandung karakter spesial di ujung kata akan dihapus. Setelah melalui kedua proses tersebut maka akan didapatkan *pure words*, siap untuk dilakukan pendeteksian kata pada *database*.

IV. HASIL DAN PEMBAHASAN

A. Pembuatan Aplikasi

Hal yang pertama dilakukan dalam membuat aplikasi adalah proses pengkodingan aplikasi. Pembuatan koding aplikasi dibuat dengan menggunakan Notepad++. Setelah menuliskan koding pada Notepad++, hal selanjutnya dalam implementasi aplikasi adalah menyimpan kode sumber yang ditulis dalam folder "TA". File ini disimpan di "C:/XAMPP/htdocs/TA".

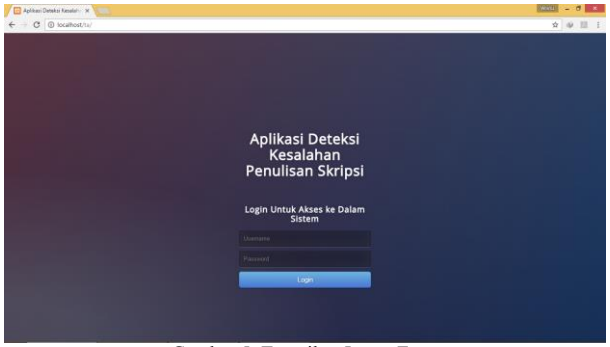
Setelah melakukan tahap pengkodingan, tahap selanjutnya dalam implementasi aplikasi adalah menjalankan Apache dan MySQL pada XAMPP untuk membuat *database* sesuai yang telah dirancang sebelumnya.

B. Ujicoba Aplikasi

Setelah aplikasi dibangun, langkah berikutnya adalah membangun dan menguji aplikasi antarmuka untuk *output* di komputer dan pengolahan *database*. Dalam tahap ini dilakukan GUI pada aplikasi yang dirancang sebelumnya.

1) *Login Form*

Ketika *user* menekan tombol *login*, maka sistem akan merespon *input value* pada kolom *username* dan *password*. Jika sesuai, maka *user* akan dipindahkan menuju *form* bab1, namun jika tidak sesuai, *user* akan kembali ke *form login*. Tampilan *Login Form* dapat dilihat pada gambar 4.



Gambar 5. Tampilan *Login Form*

2) *Form* Halaman Awal

Pada *Form* ini *user* bisa melakukan deteksi kesalahan kata dan menyimpan kata yang diinput oleh user. Ada dua buah *text box* yang harus diinput, yaitu *text* box judul bab dan *text box* isi bab. *User* harus mengisi *text box* judul bab dan isi bab sebelum melakukan deteksi kesalahan atau menyimpan data. Tampilan *Form* Halaman Awal dapat dilihat pada gambar 6.

Gambar 6. Tampilan *Form* Halaman Awal.

Gambar 7. Tampilan Hasil Deteksi Kesalahan.

Hasil dari deteksi kata yang dimasukkan adalah jika kata tersebut terdapat kesalahan, maka kata tersebut akan diberi *background* merah, mengindikasikan bahwa kata tersebut salah. Tampilan dari hasil deteksi kesalahan dapat dilihat pada gambar 7.

3) *Form* Daftar Pustaka

Form ini akan menyimpan semua data daftar pustaka yang akan disimpan oleh *user*. Tampilan *form* Daftar Pustaka dapat dilihat pada gambar 8.

Gambar 8. Tampilan Daftar Pustaka.

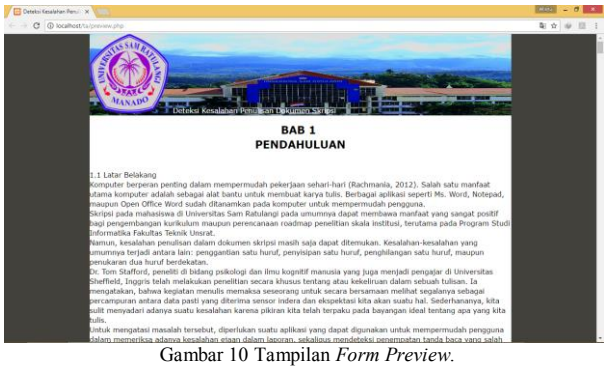
4) *Form* Gambar

Form ini berfungsi untuk menyimpan semua data gambar yang dimasukkan oleh *user*. Tampilan *form* Daftar Gambar dapat dilihat pada gambar 9.

Gambar 9. Tampilan Daftar Gambar.

5) *Form Preview*

Form ini berfungsi untuk memanggil data dari *database* yang telah disimpan oleh *user* sebelumnya pada *form* halaman awal. *Database* ini berisi teks kalimat yang dimasukkan dan disimpan oleh *user*. Tampilan *form Preview* dapat dilihat pada gambar 10.



Gambar 10 Tampilan *Form Preview*.

C. Uji Coba *Database*

Setelah aplikasi dibangun, kemudian dilakukan pembuatan *database* untuk ujicoba yang akan disambung dengan aplikasi, agar aplikasi dapat berfungsi dengan baik.

1) *Table Definition* : logindatabase

Pada tabel ini, kolom *username* diberi *Primary Key* untuk mencegah terjadinya nama *username* yang sama.

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	username	varchar(15)	latin1_swedish_ci		No	None	
2	nama	varchar(40)	latin1_swedish_ci		No	None	
3	password	varchar(30)	latin1_swedish_ci		No	None	

Gambar 11 Deskripsi Tabel logindatabase.

2) *Table Definition* : database_konten

Pada tabel ini, kolom *username* diberi *Primary Key* untuk mencegah terjadinya nama *username* yang sama pada konten, sehingga 1 *username* hanya dapat menyimpan 1 data pada tabel database_konten.

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	username	varchar(15)	latin1_swedish_ci		No	None	
2	bab1	mediumtext	latin1_swedish_ci		No	None	
3	bab2	mediumtext	latin1_swedish_ci		No	None	
4	bab3	mediumtext	latin1_swedish_ci		No	None	
5	bab4	mediumtext	latin1_swedish_ci		No	None	
6	bab5	mediumtext	latin1_swedish_ci		No	None	

Gambar 12 Deskripsi Tabel database_konten.

3) *Table Definition* : tb_katadasar

Pada tabel tb_katadasar, hanya ada 1 kolom saja, yaitu kolom kata. Kolom ini diberi *Unique Key* agar kata pada tabel ini tidak ada yang sama.

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	kata	varchar(70)	latin1_swedish_ci		No	None	

Gambar 13 Deskripsi Tabel tb_katadasar.

4) *Table Definition* : database_dp

Pada tabel ini, tidak diberi atribut apapun pada kolom tabel, karena data daftar pustaka boleh ada yang sama pada kolom tabel.

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	username	varchar(25)	latin1_swedish_ci		No	None	
2	jenis	varchar(30)	latin1_swedish_ci		No	None	
3	penulis	varchar(100)	latin1_swedish_ci		No	None	
4	judul	varchar(50)	latin1_swedish_ci		No	None	
5	judul_jurnal	varchar(100)	latin1_swedish_ci		No	None	
6	volume_jurnal	varchar(10)	latin1_swedish_ci		No	None	
7	gelar	varchar(10)	latin1_swedish_ci		No	None	
8	link	varchar(150)	latin1_swedish_ci		No	None	
9	tgl_publikasi	varchar(50)	latin1_swedish_ci		No	None	
10	tgl_akses	varchar(50)	latin1_swedish_ci		No	None	
11	instansi	varchar(100)	latin1_swedish_ci		No	None	
12	penerbit	varchar(100)	latin1_swedish_ci		No	None	

Gambar 14 Deskripsi Tabel database_dp.

5) *Table Definition* : database_gambar

Pada tabel ini, tidak ada atribut khusus seperti tabel database_dp. Hal ini dikarenakan ada data yang boleh berisi konten yang sama pada kolom tabel.

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	username	varchar(50)	latin1_swedish_ci		No	None	
2	nomor_gambar	varchar(10)	latin1_swedish_ci		No	None	
3	deskripsi_gambar	varchar(100)	latin1_swedish_ci		No	None	
4	nama_gambar	varchar(100)	latin1_swedish_ci		No	None	
5	gambar	longblob			No	None	

Gambar 15 Deskripsi Tabel database_gambar.

D. Uji Coba Sistem

Setelah Aplikasi sudah dibangun, langkah selanjutnya adalah menguji sistem pada aplikasi yang berjalan. Pengujian yang dilakukan menggunakan isi sebuah skripsi untuk melihat respon dari aplikasi yang dibangun. Tampilan dari hasil deteksi dapat dilihat pada gambar 7. Hasil dari pengujian dibuat dalam sebuah tabel *list* mengenai kesalahan yang diuji saat *user* memasukkan sebuah kata atau kalimat pada bagian tampilan halaman awal.

TABEL 6
LIST JENIS KESALAHAN YANG DIUJI

Jenis Kesalahan Yang Diuji	Keterangan
Kesalahan pada pengetikkan judul bab	Berhasil
Kesalahan kata yang tidak sesuai dengan susunan KTIS (Misal: Harusnya mengetikkan kata "1.1 Pendahuluan", bukan kata yang lain)	Berhasil
Kesalahan ejaan pada kata berbahasa Indonesia	Berhasil
Kesalahan pada spasi sebuah karakter (Misal: "kata1 . kata2")	Berhasil
Kesalahan kata yang saling sambung (tidak ada spasi)	Berhasil
Kesalahan kata yang saling sambung dengan karakter spesial (titik, koma, dll)	Berhasil
Kesalahan pada saat penyimpanan di bagian daftar gambar	Berhasil

V. PENUTUP

A. Kesimpulan

Setelah melakukan penelitian ini, dapat diambil kesimpulan bahwa aplikasi deteksi kesalahan penulisan naskah dokumen skripsi sangat membantu untuk mencari kesalahan dalam penulisan. Dengan adanya bantuan aplikasi deteksi penulisan kesalahan, mahasiswa menjadi terbantu untuk mendeteksi adanya kesalahan dalam penulisan skripsi. Penerapan aplikasi ini diharapkan dapat diimplementasikan langsung secara *real*, karena aplikasi perancangan dan pembangunan aplikasi ini telah berhasil.

B. Saran

Berdasarkan hasil penelitian dan kesimpulan dari rancang bangun aplikasi deteksi kesalahan penulisan naskah dokumen skripsi, maka terdapat beberapa saran diantaranya sebagai berikut:

1. Aplikasi yang telah dibuat dapat dimodifikasi supaya bisa digunakan pada *platform* berbentuk *mobile* agar pengguna dapat menggunakannya di mana saja dan kapan saja.
2. Aplikasi yang telah dibuat bisa dibuat dalam bahasa pemrograman lain yang lebih bagus dalam hal struktur dan fungsi.
3. Menambahkan fitur koreksi ejaan setelah dilakukan deteksi kesalahan penulisan.

DAFTAR PUSTAKA

[1] Afia, A. (2007). Kesalahan-Kesalahan Umum dalam Menulis Ilmiah. Pusat Pengembangan Bahan Ajar-UMB.

[2] Dwitiyastuti, R. (2012). Pengoreksi Kesalahan Ejaan Bahasa Indonesia Menggunakan Metode Levenshtein Distance. Teknik Elektro Fakultas Teknik Universitas Brawijaya Malang.

[3] Kimberly Martin. "Common Mistakes When Writing A Book In Microsoft Word". Internet: http://www.self-pub.net/common-mistakes-made-when-writing-a-book-in-microsoft-word/, [diakses tanggal 30 April 2017]

[4] Rahman, F. (2015) Aplikasi Pemesanan Undangan Online. Politeknik Negeri Tanah Laut.

[5] Sidik, Betha. 2014. "Pemrograman Web Dengan PHP". Penerbit Informatika. Bandung.

[6] Tim Editor. "Mengapa Kita Tidak Sadar Melakukan Typo?". Internet: https://kumparan.com/tio/mengapa-kita-tidak-sadar-melakukan-typo, 1 Maret 2017 [Diakses tanggal 22 Mei 2017]

[7] Wardani, S. (2013). Sistem Informasi Pengolahan Data Nilai Siswa Berbasis Web Pada Sekolah Menengah Atas (SMA) Muhammadiyah Pacitan. Teknik Informatika Universitas Pacitan.

[8] Widodo, M. (2016) Sistem Informasi Dan Pengolahan Data Kursus Mobil Berbasis Web Dengan Sms Gateway Di Armada Pasuruan. Universitas Merdeka Pasuruan.

SEKILAS TENTANG PENULIS

Wisnu Wardhana Adiel Umboh, lahir di Manado pada tanggal 16 Oktober 1994. Merupakan seorang mahasiswa Program Studi Informatika, Jurusan Teknik Elektro, Fakultas Teknik, Universitas Sam Ratulangi pada tahun 2012 – 2017. Selama masa kuliah, saya menjalani kerja praktek di Satpol-PP, UNSRAT, mengikuti kegiatan Kuliah Kerja Terpadu di Desa Kawangkoan, Minahasa Utara. Saya mulai menempuh pendidikan di SD Katolik 4 Manado (2000-2006). Kemudian melanjutkan ke SMP Pax Christi Manado (2006-2009). Setelah itu saya menempuh pendidikan di SMA Negeri 9 Manado (2009-2012). Setelah lulus, di tahun 2012 saya melanjutkan pendidikan di Universitas Sam Ratulangi Manado, mengambil Program Studi S-1 Teknik Informatika di Jurusan Elektro Fakultas Teknik.